

**Amendments to th Claims**

1 Claim 1 (currently amended): A document encoded in an extensible machine-oriented structured  
2 notation, wherein the document resides on one or more computer-readable media and comprises:

3 a node count representing a count of nodes in the document;

4 a node specification for each of the nodes, each of the node specifications comprising:

5 a node name;

6 a child list specifying index values of zero or more nodes which are children of the  
7 node;

8 an attribute list specifying zero or more ~~(attribute name, attribute value) pair~~ more  
9 attribute pair references for attributes of the node, each attribute pair reference comprising an  
10 attribute name and an attribute value; and

11 a node value specification, which is empty if the node has no value; and

12 a data buffer containing attribute names and attribute values referenced from the attribute  
13 lists and node values referenced from the node value specifications.

1 Claim 2 (currently amended): The document according to Claim 1, wherein each attribute pair  
2 ~~each (attribute name, attribute value) pair~~ reference specifies a starting name position, a name  
3 length, a starting value position, and a value length.

1 Claim 3 (original): The document according to Claim 2, wherein the starting name position and  
2 starting value position are relative to a beginning of the data buffer.

Serial No. 09/652,056

-11-

Docket RSW9-2000-0069-US1

1 Claim 4 (original): The document according to Claim 2, wherein the starting name position and  
2 starting value position are relative to a beginning of the document.

1 Claim 5 (original): The document according to Claim 1, wherein the node value specification  
2 specifies a starting value position and a value length.

1 Claim 6 (original): The document according to Claim 5, wherein the starting value position is  
2 relative to a beginning of the data buffer.

1 Claim 7 (currently amended): The document according to Claim 5, wherein the starting name  
2 position and starting value position is ~~is~~ ~~are~~ relative to a beginning of the document.

1 Claim 8 (currently amended): The document according to Claim 1, wherein ~~each (attribute name,~~  
2 ~~attribute value) pair~~ each attribute pair reference specifies a starting name position, an ending  
3 name position, a starting value position, and an ending value position.

1 Claim 9 (original): The document according to Claim 1, wherein the node value specification  
2 specifies a starting value position and an ending value position.

1 Claim 10 (currently amended): A computer program product embodied on one or more  
2 computer-readable media, the computer program product adapted for encoding a document in an  
3 extensible machine-oriented structured notation and comprising:

4 computer-readable program code means for encoding a node count representing a count  
5 of nodes in the document;

6 computer-readable program code means for encoding a node specification for each of the  
7 nodes, further comprising:

8 computer-readable program code means for encoding a node name;

9 computer-readable program code means for encoding a child list specifying index  
10 values of zero or more nodes which are children of the node;

11 computer-readable program code means for encoding an attribute list specifying  
12 zero or more (attribute name, attribute value) pair more attribute pair references for attributes of  
13 the node, each attribute pair reference comprising an attribute name and an attribute value; and

14 computer-readable program code means for encoding a node value specification,  
15 which is empty if the node has no value;

16 computer-readable program code means for encoding a data buffer containing attribute  
17 names and attribute values referenced from the attribute lists and node values referenced from the  
18 node value specifications; and

19 computer-readable program code means for storing the encoded node count, the encoded  
20 node specifications, and the encoded data buffer as the encoded document in memory or writing  
21 the encoded document to one or more storage media.

1 Claim 11 (currently amended): A computer program product embodied on one or more  
2 computer-readable media, the computer program product adapted for processing a document  
3 encoded in an extensible machine-oriented structured notation and comprising:

4 computer-readable program code means for parsing the document, further comprising:

5 computer-readable program code means for parsing a node count representing a  
6 count of nodes in the document;

7 computer-readable program code means for parsing a node specification for each  
8 of the nodes, further comprising:

9 computer-readable program code means for parsing a node name;

10 computer-readable program code means for parsing a child list specifying  
11 index values of zero or more nodes which are children of the node;

12 computer-readable program code means for parsing an attribute list  
13 specifying zero or more (attribute name, attribute value) pair more attribute pair references for  
14 attributes of the node, each attribute pair reference comprising an attribute name and an attribute  
15 value; and

16 computer-readable program code means for parsing a node value  
17 specification, which is empty if the node has no value; and

18 computer-readable program code means for parsing a data buffer containing  
19 attribute names and attribute values referenced from the attribute lists and node values referenced  
20 from the node value specifications; and

21 computer-readable program code means for using the parsed document as input for the  
22 processing.

1 Claim 12 (currently amended): A computer program product embodied on one or more  
2 computer-readable media, the computer program product adapted for converting an input

Serial No. 09/652,056

-14-

Docket RSW9-2000-0069-US1

3 document encoded in an extensible human-friendly extensible markup language ("XML") to an  
4 output document encoded in a machine-oriented extensible markup language ("mXML") and  
5 comprising:

6 computer-readable program code means for creating a document tree representation of the  
7 input document;

8 computer-readable program code means for obtaining a node count representing a count  
9 of nodes in the document tree representation;

10 computer-readable program code means for writing the node count to an mXML buffer;

11 computer-readable program code means for traversing each node in the document tree  
12 representation and generating a corresponding node specification in the mXML buffer, further  
13 comprising:

14 computer-readable program code means for generating a node name;

15 computer-readable program code means for generating an attribute list specifying  
16 zero or more ~~(attribute name, attribute value)~~ pair more attribute pair references for attributes of  
17 the node, each attribute pair reference comprising an attribute name and an attribute value;

18 computer-readable program code means for generating a child list specifying index  
19 values of zero or more nodes which are children of the node; and

20 computer-readable program code means for generating a node value specification,  
21 which is empty if the node has no value;

22 computer-readable program code means for generating a data buffer containing attribute  
23 names and attribute values referenced from the attribute lists and node values referenced from the  
24 node value specifications; and

25 computer-readable program code means for appending the data buffer to the mXML  
26 buffer to form the output document.

1 Claim 13 (currently amended): The computer program product according to Claim 12, wherein  
2 the computer-readable program code means for generating ~~each (attribute name, attribute value)~~  
3 ~~pair~~ each attribute pair reference further comprises computer-readable program code means for  
4 generating a starting name position, a name length, a starting value position, and a value length.

1 Claim 14 (original): The computer program product according to Claim 13, wherein the starting  
2 name position and starting value position are relative to a beginning of the data buffer.

1 Claim 15 (original): The computer program product according to Claim 13, wherein the starting  
2 name position and starting value position are relative to a beginning of the output document.

1 Claim 16 (original): The computer program product according to Claim 12, wherein the node  
2 value specification specifies a starting value position and a value length.

1 Claim 17 (original): The computer program product according to Claim 15, wherein the starting  
2 value position is relative to a beginning of the data buffer.

1 Claim 18 (currently amended): The computer program product according to Claim 15, wherein  
2 the ~~starting name position and starting value position~~ [[are]] is relative to a beginning of the

Serial No. 09/652,056

-16-

Docket RSW9-2000-0069-US1

3 output document.

1 Claim 19 (currently amended): The computer program product according to Claim 12, wherein  
2 the computer-readable program code means for generating ~~each (attribute name, attribute value)~~  
3 pair each attribute pair reference further comprises computer-readable program code means for  
4 generating a starting name position, an ending name position, a starting value position, and an  
5 ending value position.

1 Claim 20 (original): The computer program product according to Claim 12, wherein the node  
2 value specification specifies a starting value position and an ending value position.

1 Claim 21 (currently amended): A system for encoding a document in an extensible machine-  
2 oriented structured notation, comprising:

3 means for encoding a node count representing a count of nodes in the document;

4 means for encoding a node specification for each of the nodes, further comprising:

5 means for encoding a node name;

6 means for encoding a child list specifying index values of zero or more nodes

7 which are children of the node;

8 means for encoding an attribute list specifying zero or more ~~(attribute name,~~

9 ~~attribute value) pair~~ more attribute pair references for attributes of the node, each attribute pair  
10 reference comprising an attribute name and an attribute value; and

11 means for encoding a node value specification, which is empty if the node has no

Serial No. 09/652,056

-17-

Docket RSW9-2000-0069-US1

12 value;  
13 means for encoding a data buffer containing attribute names and attribute values  
14 referenced from the attribute lists and node values referenced from the node value specifications;  
15 and  
16 means for storing the encoded node count, the encoded node specifications, and the  
17 encoded data buffer as the encoded document in memory or writing the encoded document to one  
18 or more storage media.

1 Claim 22 (currently amended): A system for processing a document encoded in an extensible  
2 machine-oriented structured notation, comprising:

3 means for parsing the document, further comprising:

4 means for parsing a node count representing a count of nodes in the document;

5 means for parsing a node specification for each of the nodes, further comprising:

6 means for parsing a node name;

7 means for parsing a child list specifying index values of zero or more nodes

8 which are children of the node;

9 means for parsing an attribute list specifying zero or more ~~(attribute name,~~

10 ~~attribute value) pair~~ more attribute pair references for attributes of the node, each attribute pair

11 reference comprising an attribute name and an attribute value; and

12 means for parsing a node value specification, which is empty if the node

13 has no value; and

14 means for parsing a data buffer containing attribute names and attribute values

Serial No. 09/652,056

-18-

Docket RSW9-2000-0069-US1



15 referenced from the attribute lists and node values referenced from the node value specifications;  
16 and  
17 means for using the parsed document as input for the processing.

1 Claim 23 (currently amended): A system for converting an input document encoded in an  
2 extensible human-friendly extensible markup language ("XML") to an output document encoded  
3 in a machine-oriented extensible markup language ("mXML"), comprising:  
4 means for creating a document tree representation of the input document;  
5 means for obtaining a node count representing a count of nodes in the document tree  
6 representation;  
7 means for writing the node count to an mXML buffer;  
8 means for traversing each node in the document tree representation and generating a  
9 corresponding node specification in the mXML buffer, further comprising:  
10 means for generating a node name;  
11 means for generating an attribute list specifying zero or more ~~(attribute name;~~  
12 ~~attribute value) pair~~ more attribute pair references for attributes of the node, each attribute pair  
13 reference comprising an attribute name and an attribute value;  
14 means for generating a child list specifying index values of zero or more nodes  
15 which are children of the node; and  
16 means for generating a node value specification, which is empty if the node has no  
17 value;  
18 means for generating a data buffer containing attribute names and attribute values

19 referenced from the attribute lists and node values referenced from the node value specifications;  
20 and  
21 means for appending the data buffer to the mXML buffer to form the output document.

1 Claim 24 (currently amended): The system according to Claim 23, wherein the means for  
2 generating ~~each (attribute name, attribute value) pair~~ each attribute pair reference further  
3 comprises means for generating a starting name position, a name length, a starting value position,  
4 and a value length.

1 Claim 25 (original): The system according to Claim 24, wherein the starting name position and  
2 starting value position are relative to a beginning of the data buffer.

1 Claim 26 (original): The system according to Claim 24, wherein the starting name position and  
2 starting value position are relative to a beginning of the output document.

1 Claim 27 (original): The system according to Claim 23, wherein the node value specification  
2 specifies a starting value position and a value length.

1 Claim 28 (original): The system according to Claim 26, wherein the starting value position is  
2 relative to a beginning of the data buffer.

1 Claim 29 (currently amended): The system according to Claim 26, wherein the ~~starting name~~

Serial No. 09/652,056

-20-

Docket RSW9-2000-0069-US1

2 position and starting value position ~~[[are]]~~ is relative to a beginning of the output document.

1 Claim 30 (currently amended): The system according to Claim 23, wherein the means for  
2 generating ~~each (attribute name, attribute value) pair~~ each attribute pair reference further  
3 comprises means for generating a starting name position, an ending name position, a starting value  
4 position, and an ending value position.

1 Claim 31 (original): The system according to Claim 23, wherein the node value specification  
2 specifies a starting value position and an ending value position.

1 Claim 32 (currently amended): A method for encoding a document in an extensible machine-  
2 oriented structured notation, comprising the steps of:  
3 encoding a node count representing a count of nodes in the document;  
4 encoding a node specification for each of the nodes, further comprising the steps of:  
5 encoding a node name;  
6 encoding a child list specifying index values of zero or more nodes which are  
7 children of the node;  
8 encoding an attribute list specifying zero or more ~~(attribute name, attribute value)~~  
9 pair more attribute pair references for attributes of the node, each attribute pair reference  
10 comprising an attribute name and an attribute value; and  
11 encoding a node value specification, which is empty if the node has no value;  
12 encoding a data buffer containing attribute names and attribute values referenced from the

attribute lists and node values referenced from the node value specifications; and  
storing the encoded node count, the encoded node specifications, and the encoded data  
buffer as the encoded document in memory or writing the encoded document to one or more  
storage media.

Claim 33 (currently amended): A method for processing a document encoded in an extensible  
machine-oriented structured notation, comprising the steps of:

parsing the document, further comprising the steps of:

parsing a node count representing a count of nodes in the document;

parsing a node specification for each of the nodes, further comprising the steps of:

parsing a node name;

parsing a child list specifying index values of zero or more nodes which are  
children of the node;

parsing an attribute list specifying zero or more ~~(attribute name, attribute  
value) pair~~ more attribute pair references for attributes of the node, each attribute pair reference  
comprising an attribute name and an attribute value; and

parsing a node value specification, which is empty if the node has no value;

and

parsing a data buffer containing attribute names and attribute values referenced  
from the attribute lists and node values referenced from the node value specifications; and  
using the parsed document as input for the processing.

1 Claim 34 (currently amended): A method for converting an input document encoded in an  
2 extensible human-friendly extensible markup language ("XML") to an output document encoded  
3 in a machine-oriented extensible markup language ("mXML"), comprising the steps of:  
4 creating a document tree representation of the input document;  
5 obtaining a node count representing a count of nodes in the document tree representation;  
6 writing the node count to an mXML buffer;  
7 traversing each node in the document tree representation and generating a corresponding  
8 node specification in the mXML buffer, further comprising the steps of:  
9 generating a node name;  
10 generating an attribute list specifying zero or more ~~(attribute name, attribute value)~~  
11 pair more attribute pair references for attributes of the node, each attribute pair reference  
12 comprising an attribute name and an attribute value;  
13 generating a child list specifying index values of zero or more nodes which are  
14 children of the node; and  
15 generating a node value specification, which is empty if the node has no value;  
16 generating a data buffer containing attribute names and attribute values referenced from  
17 the attribute lists and node values referenced from the node value specifications; and  
18 appending the data buffer to the mXML buffer to form the output document.

1 Claim 35 (currently amended): The method according to Claim 34, wherein the step of  
2 generating ~~each (attribute name, attribute value) pair~~ each attribute pair reference further  
3 comprises the step of generating a starting name position, a name length, a starting value position,

4 and a value length.

1 Claim 36 (original): The method according to Claim 35, wherein the starting name position and  
2 starting value position are relative to a beginning of the data buffer.

1 Claim 37 (original): The method according to Claim 35, wherein the starting name position and  
2 starting value position are relative to a beginning of the output document.

1 Claim 38 (original): The method according to Claim 34, wherein the node value specification  
2 specifies a starting value position and a value length.

1 Claim 39 (original): The method according to Claim 37, wherein the starting value position is  
2 relative to a beginning of the data buffer.

1 Claim 40 (currently amended): The method according to Claim 37, wherein the ~~starting name~~  
2 ~~position and starting value position~~ [[are]] is relative to a beginning of the output document.

1 Claim 41 (currently amended): The method according to Claim 34, wherein the step of  
2 generating each ~~(attribute name, attribute value) pair~~ each attribute pair reference further  
3 comprises the step of generating a starting name position, an ending name position, a starting  
4 value position, and an ending value position.

1 Claim 42 (original): The method according to Claim 34, wherein the node value specification  
2 specifies a starting value position and an ending value position.

1 Claim 43 (original): A document encoded in an extensible machine-oriented structured notation,  
2 wherein the document resides on one or more computer-readable media and comprises:  
3 a node count representing a count of nodes in the document;  
4 a node specification for each of the nodes, each of the node specifications comprising:  
5 a node name;  
6 a child list specifying index values of zero or more nodes which are children of the  
7 node; and  
8 a node value specification, which is empty if the node has no value; and  
9 a data buffer containing node values referenced from the node value specifications.

1 Claim 44 (original): A method for encoding a document in an extensible machine-oriented  
2 structured notation, comprising the steps of:  
3 encoding a node count representing a count of nodes in the document;  
4 encoding a node specification for each of the nodes, further comprising the steps of:  
5 encoding a node name;  
6 encoding a child list specifying index values of zero or more nodes which are  
7 children of the node; and  
8 encoding a node value specification, which is empty if the node has no value;  
9 encoding a data buffer containing node values referenced from the node value

10 specifications; and  
11 storing the encoded node count, the encoded node specifications, and the encoded data  
12 buffer as the encoded document in memory or writing the encoded document to one or more  
13 storage media.